PATENT
Case No. N0080US

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Appl. No.	:	09/729,939
Applicant	:	Rajashri Joshi et al.
Filed	:	December 5, 2000
Titled	:	Method and System for Representation of Geographical Features in a Computer-based System

DECLARATION UNDER 37 CFR 1.131

The undersigned, OLE HENRY DORUM, hereby declares that:

1. I am a co-inventor of the invention described and claimed in the above-identified patent application.

2. On July 10, 2006, I signed a DECLARATION UNDER 37 CFR 1.131 along with co-inventors, RAJASHRI JOSHI and VIJAYA ISRANI. In that DECLARATION we indicated that before August 25, 2000, we invented a new method for representing geographic features. Part of this new method included, fitting a polynomial spline to the a geographic feature by applying a least squares approximation to data points specifying latitude and longitude coordinates to generate a plurality of control points for the polynomial spline.

3. Before August 25, 2000, a source code was developed; a redacted copy of this source code is attached hereto (Exhibit 1). The source code fits a polynomial spline to a geographic feature by applying a least squares approximation to data points specifying latitude and longitude coordinates to generate a plurality of control points for the polynomial spline. Specifically, the source code includes a function named 'uniformbsplinelsfit' that fits a polynomial spline to data points by applying a least square approximation to generate control points for the polynomial spline. Additionally, the function 'uniformbspline' plots the generated control points and the polynomial spline line.

4. Before August 25, 2000, the source code was tested and found to work. Specifically, the source code was tested and provided the output plot attached hereto (Exhibit 2). The blue points in the output plot are shape points specifying latitude and longitude coordinates of points along a road segment. The control points of the polynomial spline computed by the source code are shown as black circles in the output plot and the polynomial spline line is shown in red.

5. All statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true, and further these statements are made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code, and that such willful statements may jeopardize the validity of the application or any patent issuing thereon.

OLE HENRY DORUM

2/21/2007
Date

NAVTEQ North America, LLC
222 Merchandise Mart Plaza, Suite 900
Chicago, IL 60654
(312) 894-7000 x7365

```
function [Px, Py, Kx, Ky] =
uniformbsplinelsfit(x,y,xd,yd,t,C,N,rf,m1,mc,w)

% Draws a ls fit uniform B-spline curve given
% the data points, tangent vectors at the endpoints
% parameter vector, knot vector, number of control points,
% number of data points and regularization factor and tangent
% weight

for i = 1:C
   for m = 1:N
      Bv(i,m) = Bu(i-1,t(m));
   end;
end;

Bvd(1:4,1)     = [-0.5  0     0.5  0]';
Bvd(C-3:C,C)   = [ 0    -0.5  0    0.5]';

NodeWeight = 10000;

% Compute S, Sx, Sy
S = [redacted]; Sx = [redacted]; Sy = [redacted];
for l = 1:C
   for i = 1:C
      for m = 1:N
            S(l,i) = [redacted];
      end
      % Add extra weight to nodes %
      S(l,i) = [redacted];
     S(l,i) = [redacted];

      % Set up least squares equation matrix
      S(l,i) = [redacted]
[redacted];
   end
end
for l = 1:C
   for m = 1:N
      Sx(l) = [redacted];
      Sy(l) = [redacted];
   end
   % Add extra weight to nodes %
   Sx(l) = [redacted];
   Sy(l) = [redacted];
   Sx(l) = [redacted];
   Sy(l) = [redacted];

   % Set up least squares equation data matrix
   Sx(l) = [redacted];
   Sy(l) = [redacted];
end

[redacted]
S = [redacted];

xc = [redacted];

[Px, Py, Kx, Ky] = uniformbspline(xc,yc);
```

```
function [Px, Py, Kx, Ky] = uniformbspline(x,y)
% Evaluates the uniform cubic B-spline curve given the
% control points

MB = (1/6) * [ -1   3  -3   1;
                3  -6   3   0;
               -3   0   3   0;
                1   4   1   0];

MHA = [  2   -2   1   1;
        -3    3  -2  -1;
         0    0   1   0
         1    0   0   0];

MHB = (1/6)*[0     0    -1     1;
                 0     0     3     0;
                -6     6    -2    -1;
                 6     0     0     0];

C = length(x); Delta = 0.1; InvDelta = 1/Delta;

Px = []; Py = []; Kx = []; Ky = []; GA = []; GB = [];

for seg=1:C-3
   if (seg == C-3) i = 0:Delta:1;
   else i = 0:Delta:1-Delta;
   end;
   xt = x(seg:seg+3);   yt = y(seg:seg+3);
   Pxi=polyval(MB*xt,i); Pyi=polyval(MB*yt,i);
   Kx = [Kx; Pxi(1)]; Ky = [Ky; Pyi(1)];
   Px = [Px; Pxi']; Py = [Py; Pyi'];


   GHA = inv(MHA)*MB*[xt yt];
   GA = [GA; GHA];
   GHB = inv(MHB)*MB*[xt yt];
   GB = [GB; GHB];

end;

Kx = [Kx; Pxi(end)]; Ky = [Ky; Pyi(end)];

% Find cubic spline coefficients, type A
if 0
      x = GA(:,1); y = GA(:,2);
      C = length(x); Delta = 0.1; InvDelta = 1/Delta;
      for seg=1:4:C
         xt = x(seg:seg+3);   yt = y(seg:seg+3);
         Pxi=polyval(MHA*xt,i); Pyi=polyval(MHA*yt,i);
         plot(Pxi, Pyi, 'g-');
      end;
end;

% Find cubic spline coefficients, type B
if 0
      x = GB(:,1); y = GB(:,2);
      C = length(x); Delta = 0.1; InvDelta = 1/Delta;
      for seg=1:4:C
         xt = x(seg:seg+3);   yt = y(seg:seg+3);
         Pxi=polyval(MHB*xt,i); Pyi=polyval(MHB*yt,i);
         plot(Pxi, Pyi, 'g-'); plot(xt(1), yt(1), 'go');
      end;
end;
```